**Final Project: Approximate PPR Algorithm**

**Group 33: 106061218 李丞恩**

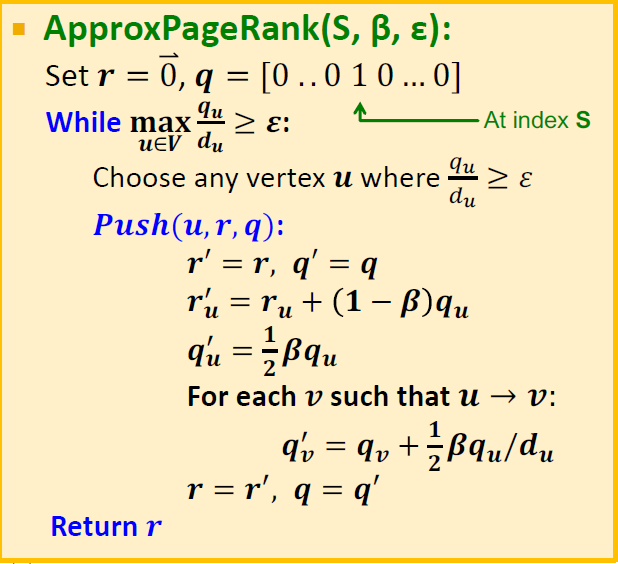
1. **Specification**

　　本專題要實作Approximate PPR Algorithm，出自講義第10章的第45頁。在將資料讀入後，會存成RDD資料結構，並選取一個頂點作為seed node。經由一系列的map-reduce運算後，得出資料集中每一點的PPR。最後，利用matplotlib繪圖，並進行資料分析與解釋。

　　我所使用的dataset是General Relativity and Quantum Cosmology collaboration network。[1]它是由Leskovc等人蒐集在arXiv網站上1993年1月到2004年1月的論文中，[2]學者間合作撰寫論文的關係而成的資料集。其中每個頂點代表一位作者，若兩名學者之間有一條邊，就代表兩人曾經合寫過論文。這個資料集包含5242個頂點，14496個邊，兩點之間沒有多重邊（multiple edges）。但某些頂點具有self edge。為了實作PPR algorithm，在讀取檔案時我排除掉了所有self edge，但由於有一個vertex所有的邊就是一條self edge，因此最後實際上只有讀入5241個頂點。

1. **Implementation**

　　我大致上依照以下的虛擬碼實作Approximate PPR演算法：



▲圖一、Approximate演算法的虛擬碼

　　其中代表點的degree，為點的PPR值，為點u的residue值，與分別為所有頂點的PPR值或residue值所形成的向量。而值設為0.8，S為seed node，則為給定的誤差值。

　　我使用Python與pyspark模組，並配合map reduce的方法進行實作。關於各個mapper與reducer的詳細設計細節，以及程式碼的解說，我都寫在ipynb檔案的markdown裡面。首先我將dataset讀入一個RDD中，接著經由一系列map-reduce，將各點初始、、以及其所連到的各個vertex存入該RDD內。接著挑選1個seed node後實作PPR演算法。

　　一開始，我的確是完全按照虛擬碼執行，不過可能出於我電腦效能實在太差的緣故，跑了數次迴圈便會出現timeout的錯誤，無法達成讓所有都變成小於的值。經過反覆驗證後發現，問題出在我每次尋找滿足

的頂點時耗費太多運算資源，我進一步發現前三次迴圈都能執行的相當快速，但第4次迴圈就會跑很久，而第5次迴圈就會timeout。因此只好改採折衷方式，只跑4次迴圈。程式的運行結果會一個資料夾，裡面的part-00000、part-00001、part-00002、part-00003由手動改為.txt檔後就是每個點與所對應的PPR值。

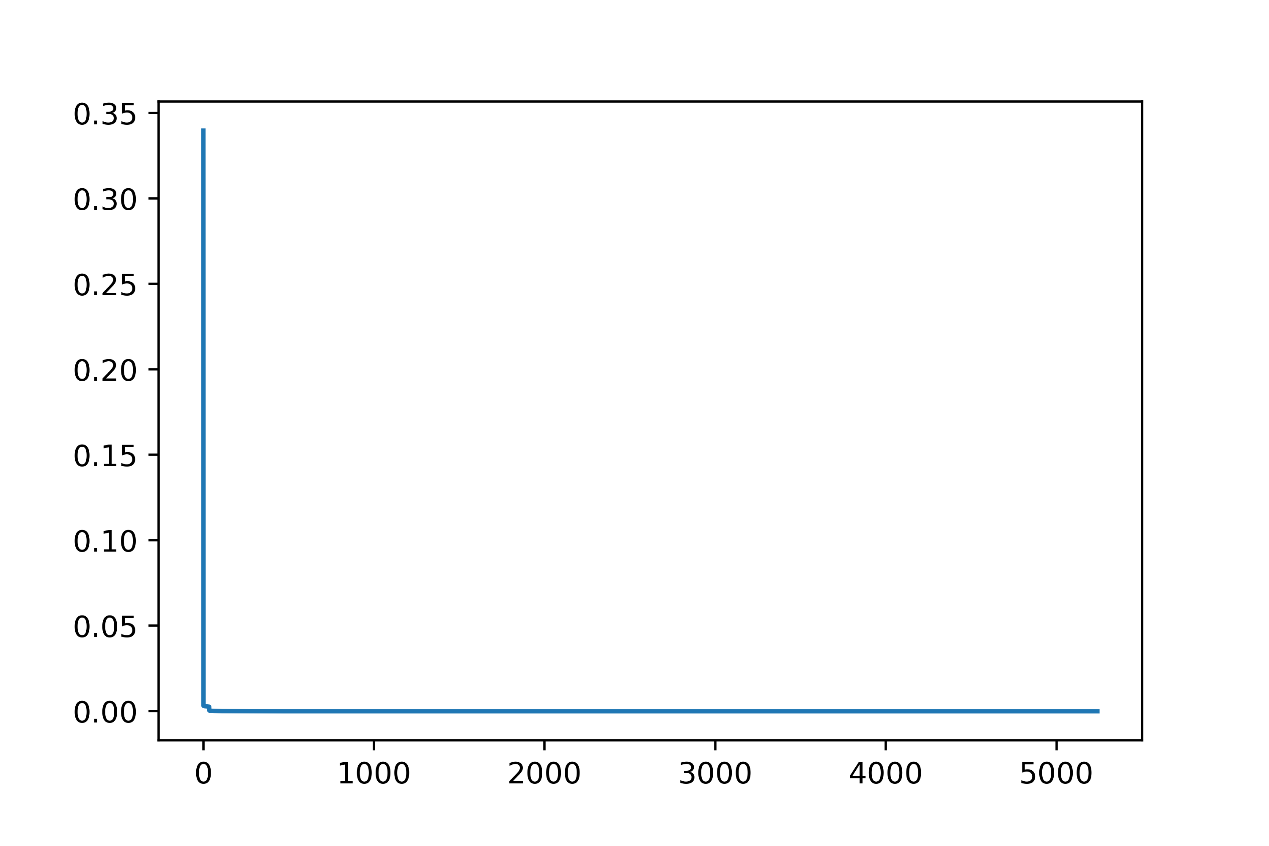
1. **Discussion**

　　只跑4次迴圈明顯是不夠的，將排序後的PPR值對排序名次畫出來的圖形會長成這樣：

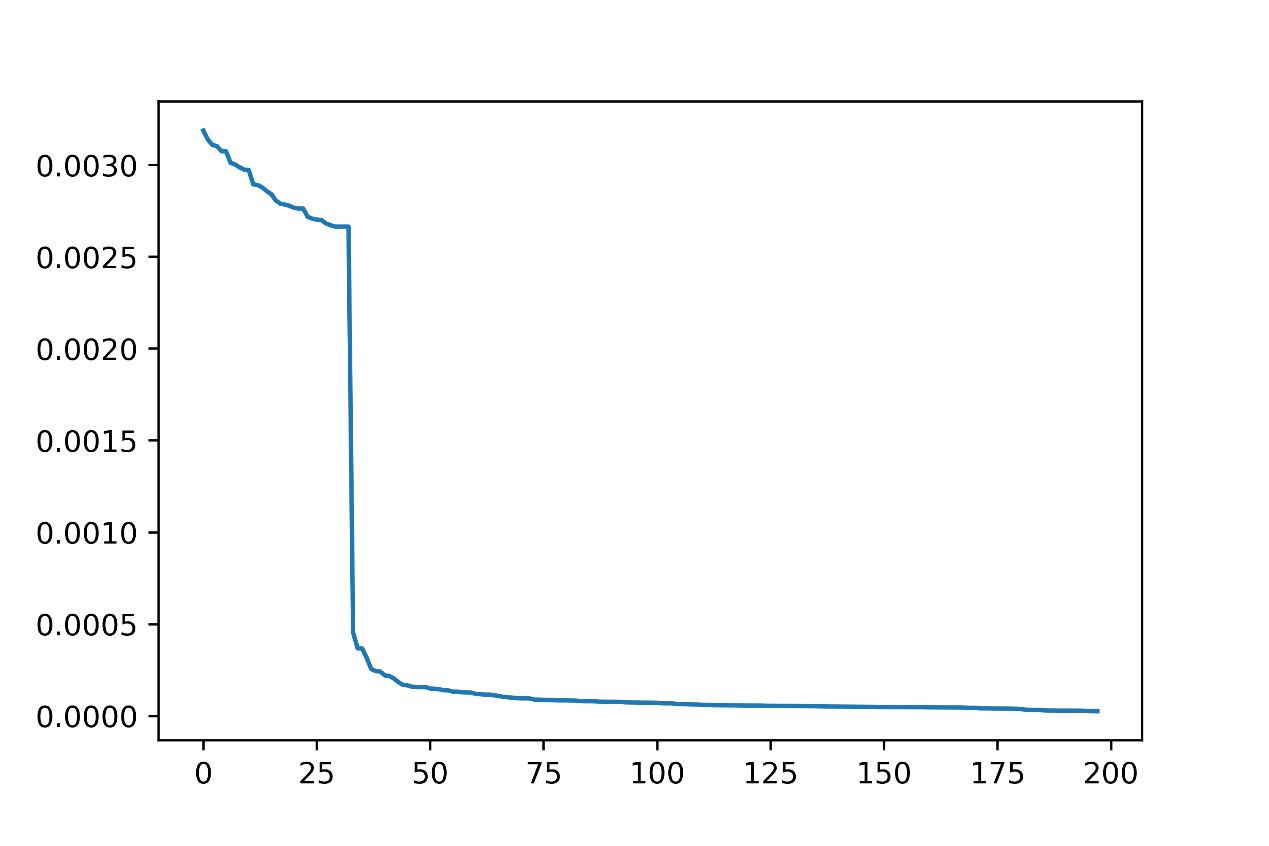


▲圖二、PPR值仍集中在seed node上

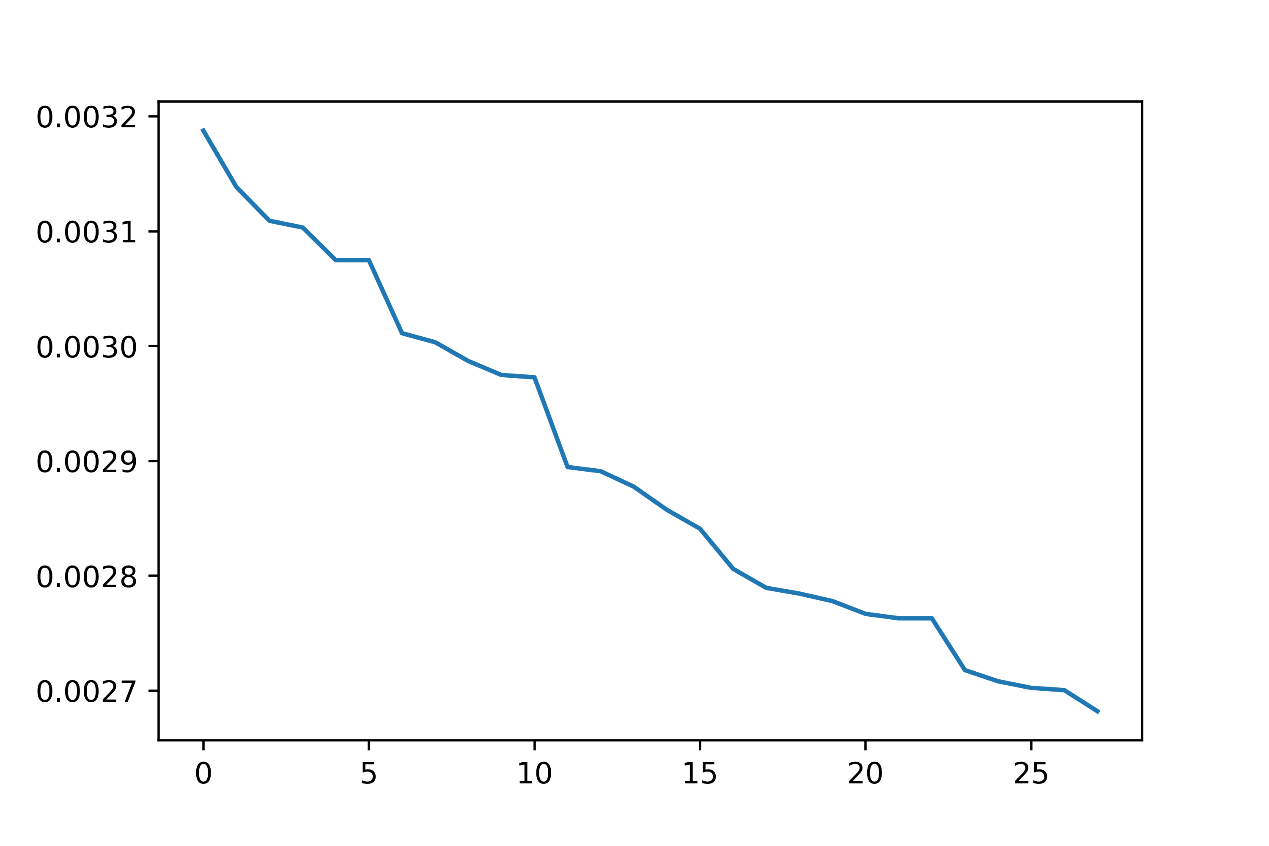
　　也就是PPR score仍集中在seed node上面，沒有擴散出去。因此我用另一個package networkX重新實作了PPR，輸出後在google colab上做圖。設定seed node為9572，值為0.8，為，得出各點PPR值後將其排序，以橫軸為名次，縱軸為PPR值做圖，得出以下結果：



▲圖三、所有vertex的PPR排序

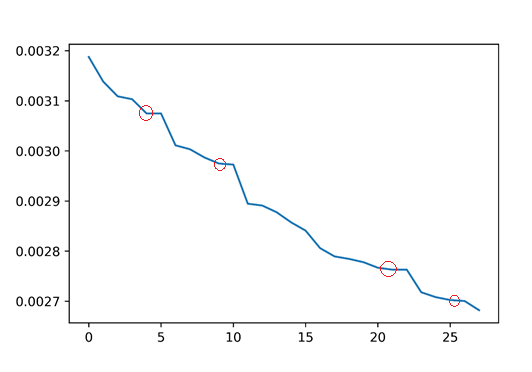


▲圖四、第2到第200名的vertex，其PPR排序



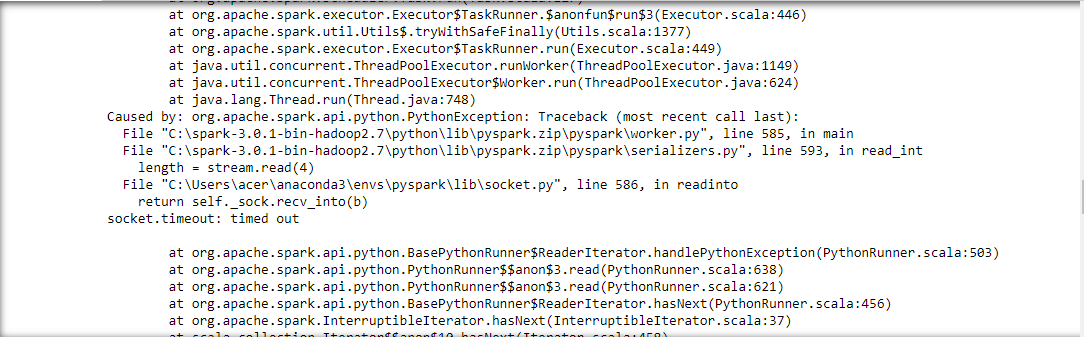
▲圖五、第2到第30名的vertex，其PPR排序

　　由上圖可知，雖然大部分的PPR值仍集中在seed node上，但是藉由lazy random walk，已經將部分的PPR分數給傳遞出去。接著，藉由觀察圖五中的local minima數量，我們可以發現這30個點大致可分為4群，如下圖所示：



▲圖六、local minima數決定分群之數量

　　而關於為何pyspark無法正常運行的問題，我曾經想過是否可能是dataset太大的所致，因此我下載了另一個非常小的dataset，例如dolphins，[3]總共只有62個頂點與159個邊，但是也發現同樣的問題，即迴圈最多只能執行4次後就會出現timeout的錯誤。



▲圖七、timeout錯誤

　　我另外嘗試了不同的解決方法：換到google colab上運行，也會出現同樣的問題，另外也在網路上搜尋相關解決方案，嘗試了增加運行的cpu數量以及sparkconf的heartbeat時間，但都無法解決我遇到的問題。我最後嘗試把迴圈內的code提出來一行一行運行，仍會發生一樣的問題。不過，若從逐行運行的結果顯示出來，仍可以發現我還是有正確地做出了Approximate PPR演算法，不過就是沒有辦法跑出一個可接受的結果。

1. **Conclusion**

　　簡單講解一下各個檔案是什麼：

|  |  |
| --- | --- |
| 檔案 | 說明 |
| CA-GrQc.txt | dataset |
| Term\_Project\_Group33.ipynb | 利用pyspark實作PPR演算法的程式 |
| PPR\_use\_networkx.ipynb | 利用networkX實作PPR演算法 |
| plot.ipynb | 繪製圖三～圖五的程式，於google colab運行（因為我的電腦跑不動） |
| part-00000.txt | Term\_Project\_Group33.ipynb的輸出 |
| part-00001.txt |
| part-00002.txt |
| part-00003.txt |
| output.txt | PPR\_use\_networkx.ipynb的輸出 |
| Term\_Project\_Group33.pdf | 本報告 |

▲表一、各個檔案的內容

1. **References**

[1] General Relativity and Quantum Cosmology collaboration network

<https://snap.stanford.edu/data/ca-GrQc.html>

[2] J. Leskovec, J. Kleinberg and C. Faloutsos. [Graph Evolution: Densification and Shrinking Diameters](http://www.cs.cmu.edu/~jure/pubs/powergrowth-tkdd.pdf). *ACM Transactions on Knowledge Discovery from Data (ACM TKDD)*, 1(1), 2007.

[3] Dolphins, <http://networkrepository.com/soc-dolphins.php>